



yocto ·
PROJECT

THE
LINUX
FOUNDATION

Hands-On Kernel Lab: Introduction to linux-yocto, kernel config fragments and common workflow patterns

Tim Orling, Intel Corporation

Yocto Project Summit *Virtual*, 2021.11

Abstract

The Linux kernel is a key component of your board support package (BSP). In this session, we will discuss various practical ways of building the Linux kernel in the Yocto Project. We will cover building a traditional git tree and defconfig, an out-of-tree kernel module, a linux-yocto based kernel, adding kernel fragments for additional functionality and other common workflow patterns.

This session will be a combination of a talk and hands-on labs.

Description

The linux-yocto workflow is a powerful and flexible way to provide a consistent kernel experience across many platforms. But the use of yocto-kernel-cache metadata (a structured tree of kernel fragments) and the linux-yocto git repository often confuses newcomers to the Yocto Project. Many traditional developers prefer to use “a git tree and a defconfig” to build their kernel, so we will also cover this use case. Individual platforms will also differ in the use of bootloader, device tree and other details that involve where the kernel is installed and how it is booted. We will give hands-on practical examples of these use cases to help you on your journey to creating and working with well-behaved Yocto Project BSP layers.

Previous Hands-On Kernel Presentations

- [Yocto Project Dev Day Virtual 2020 #3: Yocto Project Kernel Lab, Hands-On, Part 1](#) -- Trevor Woerner
- [Yocto Project Dev Day Virtual 2020 #3: Yocto Project Kernel Lab, Hands-On, Part 2](#) -- Trevor Woerner
- [Live Coding with Yocto Project #6: kernel handling and development](#) -- Josef Holtzmayr (The Yocto Jester)
- [Live Coding with Yocto Project #13: Building an out of tree kernel module](#) -- Josef Holtzmayr (The Yocto Jester)
- ["linux-yocto reference kernel maintenance and kernel workflows"](#) -- Bruce Ashfield
- [Working with the Linux Kernel in the Yocto Project](#) -- Sean Hudson
- Not an exhaustive list, there are more.

Agenda

- **Why linux-yocto?**
- **Why kernel config fragments?**
 - yocto-kernel-cache aka KMETA
- **“Traditional” kernel developer workflows**
 - mainline kernel and defconfig
 - local git tree and defconfig
 - patch series with quilt
- **Yocto Project kernel best practices**
 - linux-yocto with config fragments and patches

Agenda

- **Hands-on Lab Exercises**
 - Lab #1: Mainline kernel and menuconfig
 - Lab #2: linux-yocto with config fragments and patches
 - Lab #3: Custom Kernel recipe
 - Lab #4: Custom Kernel recipe with local git tree
- **Future Labs on device tree workflows**
- **Future Labs on creating and working with well-behaved BSP layers**



Why linux-yocto?

Why linux-yocto?

- Curated git tree
- Branches for SoCs and kernel versions
- Tested on supported SoCs
- Provides tooling to help protect you from bad configurations
- Provides tooling to help you maintain multiple kernel versions for multiple SoCs/MACHINES/boards.

What is linux-yocto?

<https://git.yoctoproject.org/linux-yocto/>

The screenshot shows the 'index : linux-yocto' page of the Yocto Project's Git repository. The page has a dark header with the 'yocto PROJECT' logo and 'index : linux-yocto' text. Below the header, there are navigation tabs: 'about', 'summary' (selected), 'refs', 'log', 'tree', 'commit', 'diff', and 'stats'. A 'log msg' dropdown and a 'search' button are also present. The main content area is divided into three columns: 'Branch', 'Commit message', and 'Author'. The 'Branch' column lists various branches like 'v5.10/standard/xilinx-zynqmp' and 'v5.10/standard/x86'. The 'Commit message' column shows the corresponding merge messages. The 'Author' column lists the authors, mostly 'Bruce Ashfield'. Below this, there is a 'Tag' section with a list of tags and their corresponding download links. The 'Download' section lists the download links for various versions. The 'Age' section shows the age of the tags. The 'Commit message' section shows the commit message for the 'HEAD' and 'master' branches. The 'Author' section shows the author of the commit.

Branch	Commit message	Author	Age
v5.10/standard/xilinx-zynqmp	Merge branch 'v5.10/standard/base' into v5.10/standard/xilinx-zynqmp	Bruce Ashfield	4 days
v5.10/standard/x86	Merge branch 'v5.10/standard/base' into v5.10/standard/x86	Bruce Ashfield	4 days
v5.10/standard/tiny/x86	Merge branch 'v5.10/standard/base' into v5.10/standard/tiny/x86	Bruce Ashfield	4 days
v5.10/standard/tiny/common-pc	Merge branch 'v5.10/standard/base' into v5.10/standard/tiny/common-pc	Bruce Ashfield	4 days
v5.10/standard/tiny/base	Merge branch 'v5.10/standard/base' into v5.10/standard/tiny/base	Bruce Ashfield	4 days
v5.10/standard/tiny/arm-versatile-926ejs	Merge branch 'v5.10/standard/base' into v5.10/standard/tiny/arm-versatile-926ejs	Bruce Ashfield	4 days
v5.10/standard/ti-sdk-5.10/ti-j72xx	Merge branch 'v5.10/standard/base' into v5.10/standard/ti-sdk-5.10/ti-j72xx	Bruce Ashfield	4 days
v5.10/standard/ti-am335x	Merge branch 'v5.10/standard/base' into v5.10/standard/ti-am335x	Bruce Ashfield	4 days
v5.10/standard/sdkv5.4/lnx-soc	Merge branch 'v5.10/standard/base' into v5.10/standard/sdkv5.4/lnx-soc	Bruce Ashfield	4 days
v5.10/standard/sdkv5.10/lnx-soc	Merge branch 'v5.10/standard/base' into v5.10/standard/sdkv5.10/lnx-soc	Bruce Ashfield	4 days
[...]			

Tag	Download	Author	Age
v5.15.2	linux-yocto-5.15.2.zip linux-yocto-5.15.2.tar.gz linux-yocto-5.15.2.tar.bz2	Greg Kroah-Hartman	2 weeks
v5.14.18	linux-yocto-5.14.18.zip linux-yocto-5.14.18.tar.gz linux-yocto-5.14.18.tar.bz2	Greg Kroah-Hartman	2 weeks
v5.10.79	linux-yocto-5.10.79.zip linux-yocto-5.10.79.tar.gz linux-yocto-5.10.79.tar.bz2	Greg Kroah-Hartman	2 weeks
v5.4.159	linux-yocto-5.4.159.zip linux-yocto-5.4.159.tar.gz linux-yocto-5.4.159.tar.bz2	Greg Kroah-Hartman	2 weeks
v5.15.1	linux-yocto-5.15.1.zip linux-yocto-5.15.1.tar.gz linux-yocto-5.15.1.tar.bz2	Greg Kroah-Hartman	3 weeks
v5.14.17	linux-yocto-5.14.17.zip linux-yocto-5.14.17.tar.gz linux-yocto-5.14.17.tar.bz2	Greg Kroah-Hartman	3 weeks
v5.10.78	linux-yocto-5.10.78.zip linux-yocto-5.10.78.tar.gz linux-yocto-5.10.78.tar.bz2	Greg Kroah-Hartman	3 weeks
v5.4.158	linux-yocto-5.4.158.zip linux-yocto-5.4.158.tar.gz linux-yocto-5.4.158.tar.bz2	Greg Kroah-Hartman	3 weeks
v5.14.16	linux-yocto-5.14.16.zip linux-yocto-5.14.16.tar.gz linux-yocto-5.14.16.tar.bz2	Greg Kroah-Hartman	3 weeks
v5.10.77	linux-yocto-5.10.77.zip linux-yocto-5.10.77.tar.gz linux-yocto-5.10.77.tar.bz2	Greg Kroah-Hartman	3 weeks
[...]			

Age	Commit message	Author
2021-11-03	Merge branch 'master-5.15' HEAD master	Bruce Ashfield

What is linux-yocto?

- Curated kernel recipes

```
meta/recipes-kernel/linux/  
├── kernel-devsrc.bb  
├── linux-dummy  
│   └── COPYING.GPL  
├── linux-dummy.bb  
├── linux-yocto-dev.bb  
├── linux-yocto-rt_5.10.bb  
├── linux-yocto-rt_5.14.bb  
├── linux-yocto-tiny_5.10.bb  
├── linux-yocto-tiny_5.14.bb  
├── linux-yocto.inc  
├── linux-yocto_5.10.bb  
└── linux-yocto_5.14.bb
```

What is linux-yocto?

- Curated kernel bbclasses

meta/classes/

...

|— devicetree.bbclass

...

|— kernel-arch.bbclass

|— kernel-artifact-names.bbclass

|— kernel-devicetree.bbclass

|— kernel-fitimage.bbclass

|— kernel-grub.bbclass

|— kernel-module-split.bbclass

|— kernel-uboot.bbclass

|— kernel-uimage.bbclass

|— kernel-yocto.bbclass

|— kernel.bbclass

|— kernelsrc.bbclass

...

|— linux-dummy.bbclass

|— linux-kernel-base.bbclass

|— module-base.bbclass

|— module.bbclass

...



Why kernel fragments?

...and what is yocto-kernel-cache?

Why kernel fragments?

- Fully supported in the Linux kernel “It’s normal”
- Modular approach to configuring the kernel
- Allows base kernel configurations to be re-used
- Allows SoC-base configurations to be maintained for MANY SoCs

What is yocto-kernel-cache?

<https://git.yoctoproject.org/yocto-kernel-cache/>

The screenshot shows the Git repository page for yocto-kernel-cache. The page has a dark header with the Yocto Project logo and the text "index : yocto-kernel-cache". Below the header, there are tabs for "summary", "refs", "log", "tree", "commit", "diff", and "stats". The "summary" tab is selected. The main content area displays a list of commits with columns for Branch, Commit message, Author, and Age. The current branch is "master".

Branch	Commit message	Author	Age
master	version: updating config/patches for v5.16	Bruce Ashfield	3 days
yocto-5.15	cfg/remoteproc: Modify REMOTEPROC from 'M' to 'Y'	XiaoLei Wang	10 days
yocto-5.14	cfg/remoteproc: Modify REMOTEPROC from 'M' to 'Y'	XiaoLei Wang	10 days
yocto-5.10	cfg/remoteproc: Modify REMOTEPROC from 'M' to 'Y'	XiaoLei Wang	10 days
yocto-5.4	kver: bumping to v5.4.159	Bruce Ashfield	11 days
yocto-5.13	README: Update mail address	Jon Mason	2 months
yocto-4.14	kver: bumping to v4.14.244	Bruce Ashfield	3 months
yocto-4.19	kver: bumping to v4.19.204	Bruce Ashfield	3 months
yocto-5.8	cfg: declare CONFIG_DRM_KMS_HELPER non-hardware	Bruce Ashfield	11 months
yocto-5.2	kver: bumping to v5.2.60	Bruce Ashfield	14 months
[...]			

Age	Commit message	Author
3 days	version: updating config/patches for v5.16 HEAD master	Bruce Ashfield
10 days	cfg/remoteproc: Modify REMOTEPROC from 'M' to 'Y'	XiaoLei Wang
11 days	beaglebone: Enable the new cpsw switch drv	Kevin Hao
2021-11-03	standard: drop CONFIG_MANDATORY_FILE_LOCKING	Bruce Ashfield
2021-11-02	kver: bump to v5.15	Bruce Ashfield
2021-10-20	yaffs: replace IS_ERR with IS_ERR_OR_NULL to check both ERR and NULL	Bruce Ashfield
2021-10-18	common-pc/qemux86*: set CONFIG_ATA_PIIX as built-in	Bruce Ashfield
2021-10-15	hardware classification: update classifications for 5.14+	Bruce Ashfield
2021-10-13	kver: bump to v5.15-rc5	Bruce Ashfield
2021-10-12	features/media: remove configs for drivers in staging	Anuj Mittal
[...]		

Clone

Structure of yocto-kernel-cache

yocto-kernel-cache

- 00-README
- arch
- backports
- bsp
- cfg
- cgl
- COPYING.GPLv2
- COPYING.MIT
- features
- kern-features.rc
- ktypes
- kver
- patches
- scripts
- small
- staging

cfg

- 8250.cfg
- 8250.scc
- ...
- fs
- ...
- net
- ...
- timer
- ...

features

- 6lowpan
- apparmor
- aufs
- bfq
- blktrace
- bluetooth
- bpf
- bsdjail
- can
- cgroups
- ciphers
- ...

.cfg and .scc files

.cfg: The config fragments

```
$ cat cfg/8250.cfg
# SPDX-License-Identifier: MIT
CONFIG_TTY=y
CONFIG_SERIAL_8250=y
CONFIG_SERIAL_8250_CONSOLE=y
CONFIG_SERIAL_8250_PCI=y
CONFIG_SERIAL_8250_NR_UARTS=4
CONFIG_SERIAL_8250_RUNTIME_UARTS=4
CONFIG_SERIAL_CORE=y
CONFIG_SERIAL_CORE_CONSOLE=y
CONFIG_SERIAL_OF_PLATFORM=y
```

.scc: “Series Configuration Control” (metadata)

```
$ cat cfg/8250.scc
# SPDX-License-Identifier: MIT
define KFEATURE_DESCRIPTION "Enable
8250 serial support"
define KFEATURE_COMPATIBILITY board

kconf hardware 8250.cfg
```




“Traditional” kernel workflows

“Traditional” Kernel Developer Workflows

- mainline kernel and defconfig
 - “I just want to use a mainline kernel and a defconfig”
 - Commonly a vanilla tarball from kernel.org
 - Commonly a full “defconfig” edited with menuconfig.
 - We’ll use this workflow in Lab #1.

“Traditional” Kernel Developer Workflows

- local git tree and defconfig
 - “I just want to use a git tree and a defconfig”
 - Commonly a local git tree
 - Commonly a full “defconfig” edited with menuconfig
 - We’ll use this workflow in Lab #3
 - Future Lab to add more tooling for “git describe” versioning

“Traditional” Kernel Developer Workflows

- patch series with quilt (and defconfig)
 - “I just want to use a quilt patch series and a defconfig”
 - Commonly (re)based on top of a vanilla mainline git tree
 - Commonly a full “defconfig” edited with menuconfig.
 - See [Using Quilt in your Workflow](#)
 - This will be a future Lab activity.



Yocto Project kernel best practices

linux-yocto with config fragments and patches

Yocto Project Kernel Best Practices

- Don't create an "evil vendor kernel"
- Unless absolutely impossible, base your kernel on linux-yocto (with patches)
- Don't use a full defconfig. Use kernel config fragments based on top of yocto-kernel-cache
- Don't create an "evil vendor kernel"
- Don't create an "evil vendor kernel"
- We'll investigate this workflow in Labs #2 and #4.



Hands-on Lab Exercises



Hands-on Kernel Lab #1

Exercise #1

Working with kernel.org tarball and menuconfig

Lab #1

- Getting to know the environment
 - ~/yp-summit-nov-21/poky/meta-lab1-qemuarm64

meta-lab1-qemuarm64/

```
|— conf
|   |— layer.conf
|   |— machine
|       |— lab1-qemuarm64.conf
|— recipes-kernel
|   |— linux
|       |— linux-korg
|           |— arm64_defconfig
|           |— defconfig
|           |— qemuarm64_defconfig
|           |— yocto-testmod.patch
|— linux-korg_5.14.bb
```

linux-korg_5.14.bb

~/yp-summit-nov-21/poky/meta-lab1-qemuarm64/recipes-kernel/linux/linux-korg_5.14.bb

```
DESCRIPTION = "Mainline Linux Kernel"
SECTION = "kernel"
LICENSE = "GPLv2"

FILESEXTRAPATHS:prepend := "${THISDIR}/files:"

LIC_FILES_CHKSUM = "file://COPYING;md5=6bc538ed5bd9a7fc9398086aedcd7e46"

inherit kernel

SRC_URI = "${KERNELORG_MIRROR}/linux/kernel/v5.x/linux-${PV}.tar.xz;name=kernel \
          file://defconfig "

S = "${WORKDIR}/linux-${PV}"

#SRC_URI += "file://yocto-testmod.patch"

SRC_URI[kernel.sha256sum] = "7e068b5e0d26a62b10e5320b25dce57588cbbc6f781c090442138c9c9c3271b2"
```

machine/lab1-qemu.conf

~/yp-summit-nov-21/poky/meta-lab1-qemuarm64/conf/machine/lab1-qemuarm64.conf

```
#@TYPE: Machine
#@NAME: lab1-qemuarm64

#@DESCRIPTION: Machine configuration for lab1-qemuarm64 systems

PREFERRED_PROVIDER_virtual/kernel ?= " linux-korg "

PREFERRED_PROVIDER_virtual/xserver ?= "xserver-xorg"
PREFERRED_PROVIDER_virtual/libgl ?= "mesa"
PREFERRED_PROVIDER_virtual/libgles1 ?= "mesa"
PREFERRED_PROVIDER_virtual/libgles2 ?= "mesa"

require conf/machine/include/qemu.inc
require conf/machine/include/ arm/armv8a/tune-cortexa57.inc

KERNEL_IMAGETYPE = "Image"

UBOOT_MACHINE ?= "qemu_arm64_defconfig"
...
```

Setup our build environment

- `$ cd ~/yp-summit-nov-21/poky`
- `$. oe-init-build-env build-kernel/`
- `$ vim conf/local.conf`
- uncomment `#MACHINE = "lab1-qemuarm64"`
- save and exit vim (`:wq`)
- `$ bitbake-layers add-layer ../meta-lab1-qemuarm64`
- `$ cat conf/bblayers.conf`

```
BBLAYERS ?= " \  
    /home/<user>/yp-summit-nov-21/poky/meta \  
    /home/<user>/yp-summit-nov-21/poky/meta-poky \  
    /home/<user>/yp-summit-nov-21/poky/meta-yocto-bsp \  
    /home/<user>/yp-summit-nov-21/poky/meta-lab1-qemuarm64 \  
    "
```

Lab #1 -- Build and Boot the Image

- `bitbake core-image-base`
- `runqemu slirp nographic`
`tmp/deploy/images/lab1-qemuarm64/Image-lab1-qemuarm64`
`.bin`
`tmp/deploy/images/lab1-qemuarm64/core-image-base-lab1`
`-qemuarm64.ext4`
 - `slirp`: user space networking (no elevated privileges required)
 - `nographic`: run in console, do not launch a GUI window
 - `path to kernel image`
 - `path to rootfs`

kernel panic!

Where's ext4?

```
[ 4.878812] No filesystem could mount root, tried:
[ 4.878893] vfat
[ 4.880105] btrfs
[ 4.880345]
[ 4.881325] Kernel panic - not syncing: VFS: Unable to mount root fs on unknown-block(253,0)
[ 4.882677] CPU: 0 PID: 1 Comm: swapper/0 Not tainted 5.14.0 #1
[ 4.884341] Hardware name: linux,dummy-virt (DT)
[ 4.885727] Call trace:
[ 4.886056] dump_backtrace+0x0/0x19c
[ 4.886676] show_stack+0x24/0x30
[ 4.887152] dump_stack_lvl+0x7c/0xa0
[ 4.887711] dump_stack+0x18/0x34
[ 4.888170] panic+0x178/0x344
[ 4.888618] mount_block_root+0x244/0x260
...
[ 4.892573] SMP: stopping secondary CPUs
[ 4.894214] Kernel Offset: disabled
[ 4.894656] CPU features: 0x11000051,000000842
[ 4.895686] Memory Limit: 256 MB
[ 4.897210] ---[ end Kernel panic - not syncing: VFS: Unable to mount root fs on
unknown-block(253,0) ]---
```

Ctrl-A X

Inspect the defconfig

```
grep -R EXT4  
~/yp-summit-nov-21/poky/meta-lab1-qemuarm64/recipes-kernel/linux/linux-korg/defconfig  
  
# CONFIG_EXT4_FS is not set
```



Lab #1 -- Configure the Kernel

PREFERRED_PROVIDER:virtual/kernel = "linux-korg"

- `bitbake -c menuconfig virtual/kernel`

```
linux-korg Configuration
config - Linux kernel 4.14.0 Kernel Configuration
> File systems
File systems
Arrow keys navigate the menu. <Enter> selects submenus ---> (or empty
submenus ----). Highlighted letters are hotkeys. Pressing <Y>
includes, <N> excludes, <M> modularizes features. Press <Esc><Esc> to
exit, <?> for Help, </> for Search. Legend: [*] built-in [ ]

[ ] Validate filesystem parameter description
< > Second extended fs support
< > The Extended 3 (ext3) filesystem
[*] The Extended 4 (ext4) filesystem
[*] Use ext4 for ext2 file systems (NEW)
[*] Ext4 POSIX Access Control Lists
[*] Ext4 Security Labels
[ ] Ext4 debugging support
[ ] JBD2 (ext4) debugging support (NEW)
< > Reiserfs support
v(+)
```

< select > < Exit > < Help > < Save > < Load >

Lab #1 -- Rebuild (only) the Kernel

```
PREFERRED_PROVIDER:virtual/kernel = "linux-korg"
```

- `bitbake -c compile -f virtual/kernel`
 - This forces the “do_compile” task to be rerun.
- `bitbake -c deploy virtual/kernel`
 - This deploys the kernel image and artifacts to the `tmp/deploy/images/lab1-qemuarm64` directory.

```
MACHINE = "lab1-qemuarm64"
```

Lab #1 -- Boot the Image

- `runqemu slirp nographic`
`tmp/deploy/images/lab1-qemuarm64/Image-lab1-qemuarm64`
`.bin`
`tmp/deploy/images/lab1-qemuarm64/core-image-base-lab1`
`-qemuarm64.ext4`
 - `slirp`: user space networking (no elevated privileges required)
 - `nographic`: run in console, do not launch a GUI window
 - `path to kernel image`
 - `path to rootfs`

Success!

```
[ 10.204580] EXT4-fs (vda): re-mounted. Opts: (null). Quota mode: disabled.
```

```
Poky (Yocto Project Reference Distro) 3.4 lab1-qemuarm64 /dev/ttyAMA0
```

```
lab1-qemuarm64 login: root
```

```
root@lab1-qemuarm64:~# uname -a
```

```
Linux lab1-qemuarm64 5.14.0 #1 SMP PREEMPT Sun Aug 29 22:04:50 UTC 2021 aarch64 GNU/Linux
```

```
root@lab1-qemuarm64:~#
```



diff .config defconfig

the .config that menuconfig
saved (our build .config)

```
$ diff tmp/work/lab1_qemuarm64-poky-linux/linux-korg/5.14-r0/ build/.config
tmp/work/lab1_qemuarm64-poky-linux/linux-korg/5.14-r0/ defconfig
3437,3444c3437
< CONFIG_EXT4_FS=y
< CONFIG_EXT4_USE_FOR_EXT2=y
< CONFIG_EXT4_FS_POSIX_ACL=y
< CONFIG_EXT4_FS_SECURITY=y
< # CONFIG_EXT4_DEBUG is not set
< CONFIG_JBD2=y
< # CONFIG_JBD2_DEBUG is not set
< CONFIG_FS_MBCACHE=y
---
> # CONFIG_EXT4_FS is not set
3466d3458
< CONFIG_FS_ENCRYPTION_ALGS=y
```

copy of our recipe's
defconfig

To share the change (make it 'permanent')

```
$ cp  
tmp/work/lab1_gemuarm64-poky-linux/linux-korg/5.14-r0/build/  
.config  
~/yp-summit-nov-21/poky/meta-lab1-gemuarm64/recipes-kernel/l  
inux/linux-korg/defconfig
```



Hands-on Kernel Lab #1

Exercise #2

Patch the kernel to add a custom kernel module

linux-korg_5.14.bb: uncomment yocto-testmod.patch

~/yp-summit-nov-21/poky/meta-lab1-qemuarm64/recipes-kernel/linux/linux-korg_5.14.bb

```
DESCRIPTION = "Mainline Linux Kernel"
SECTION = "kernel"
LICENSE = "GPLv2"

FILESEXTRAPATHS:prepend := "${THISDIR}/files:"

LIC_FILES_CHKSUM = "file://COPYING;md5=6bc538ed5bd9a7fc9398086aedcd7e46"

inherit kernel

SRC_URI = "${KERNELORG_MIRROR}/linux/kernel/v5.x/linux-${PV}.tar.xz;name=kernel \
          file://defconfig "

S = "${WORKDIR}/linux-${PV}"

SRC_URI += "file://yocto-testmod.patch"

SRC_URI[kernel.sha256sum] = "7e068b5e0d26a62b10e5320b25dce57588cbbbc6f781c090442138c9c9c3271b2"
```

yocto-testmod.patch

~/yp-summit-nov-21/poky/meta-lab1-qemuarm64/recipes-kernel/linux/linux-korg/yocto-testmod.patch

- **Add module to Kconfig**

```
+config YOCTO_TESTMOD
+    tristate "Yocto Test Driver"
+    help
+        This driver provides a silly message for testing Yocto.
+
```

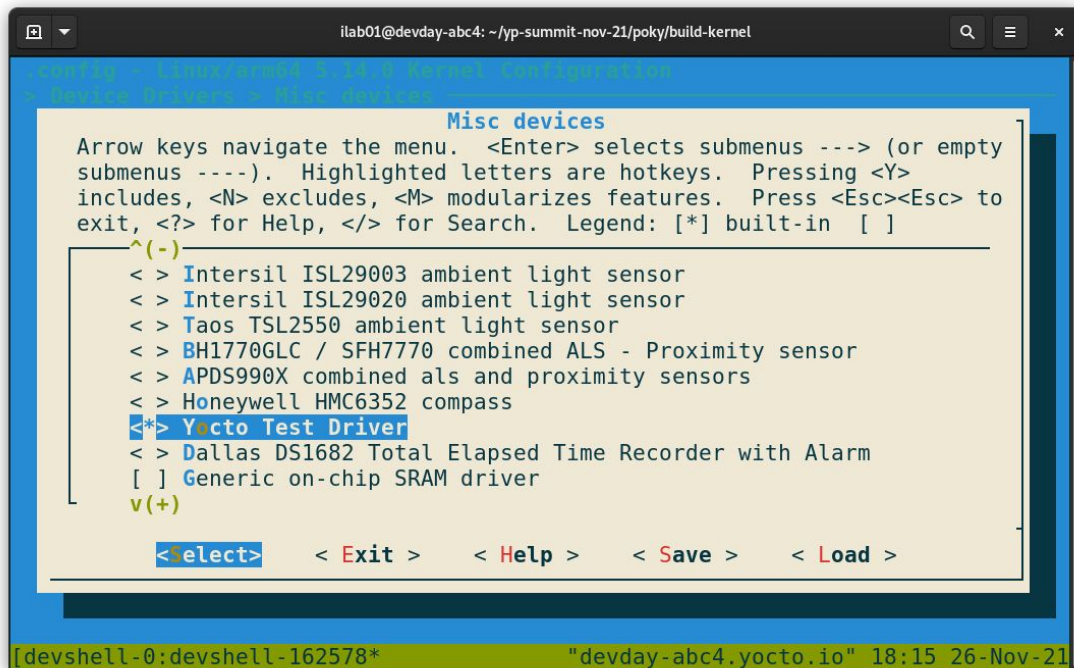
- **Add Makefile**
- **Add module init code (printk)**

```
+#include <linux/module.h>
+
+static int __init yocto_testmod_init(void)
+{
+    printk("Kilroy was here!");
+
+    return 0;
+}
```


Lab #1 -- Configure the Kernel

PREFERRED_PROVIDER:virtual/kernel = "linux-korg"

- `bitbake -c menuconfig virtual/kernel`



```
ilab01@devday-abc4: ~/yp-summit-nov-21/poky/build-kernel
.config - Linux kernel 5.14.0 Kernel Configuration
> Device Drivers > Misc devices

                                Misc devices
Arrow keys navigate the menu. <Enter> selects submenus ---> (or empty
submenus ----). Highlighted letters are hotkeys. Pressing <Y>
includes, <N> excludes, <M> modularizes features. Press <Esc><Esc> to
exit, <?> for Help, </> for Search. Legend: [*] built-in [ ]

^(-)
< > Intersil ISL29003 ambient light sensor
< > Intersil ISL29020 ambient light sensor
< > Taos TSL2550 ambient light sensor
< > BH1770GLC / SFH7770 combined ALS - Proximity sensor
< > APDS990X combined als and proximity sensors
< > Honeywell HMC6352 compass
[*] Yocto Test Driver
< > Dallas DS1682 Total Elapsed Time Recorder with Alarm
[ ] Generic on-chip SRAM driver
v(+)

<Select>  < Exit >  < Help >  < Save >  < Load >

[devshell-0:devshell-162578* "devday-abc4.yocto.io" 18:15 26-Nov-21]
```

Lab #1 -- Rebuild (only) the Kernel

- `bitbake -c compile -f virtual/kernel`
 - This forces the “do_compile” task to be rerun.
 - This is actually now optional since the recipe is already “tainted”.
- `bitbake -c deploy virtual/kernel`
 - This deploys the kernel image and artifacts to the `tmp/deploy/images/lab1-qemuarm64` directory.

Lab #1 -- Boot the Image

“Up” arrow to reuse the command (it hasn’t changed) or “Ctrl-R” and start typing for bash completion.

- `runqemu slirp nographic`
`tmp/deploy/images/lab1-qemuarm64/Image-lab1-qemuarm64`
`.bin`
`tmp/deploy/images/lab1-qemuarm64/core-image-base-lab1`
`-qemuarm64.ext4`
 - `slirp`: user space networking (no elevated privileges required)
 - `nographic`: run in console, do not launch a GUI window
 - `path to kernel image`
 - `path to rootfs`

Success!

If you get a kernel panic, also make and save the EXT4 File System changes we made earlier.

```
[ 5.161305] EXT4-fs (vda): re-mounted. Opts: (null). Quota mode: disabled.
```

```
Poky (Yocto Project Reference Distro) 3.4 lab1-qemuarm64 /dev/ttyAMA0
```

```
lab1-qemuarm64 login: root
```

```
root@lab1-qemuarm64:~# dmesg | grep Kilroy
```

```
[ 2.428027] Kilroy was here!
```

```
root@lab1-qemuarm64:~#
```

vo/

To share the change (make it 'permanent')

```
$ cp  
tmp/work/lab1_gemuarm64-poky-linux/linux-korg/5.14-r0/build/  
.config  
~/yp-summit-nov-21/poky/meta-lab1-gemuarm64/recipes-kernel/l  
inux/linux-korg/defconfig
```

A decorative pattern of dark gray hexagons arranged in a staggered grid, located in the top-left corner of the slide.

Lab#1 Complete!

yocto ·
PROJECT

THE
LINUX
FOUNDATION



Hands-on Kernel Lab #2

Exercise #1

Working with linux-yocto and config fragments

Lab #2

- Getting to know the environment
 - ~/yp-summit-nov-21/poky/meta-lab2-qemuarm64

meta-lab2-qemuarm64/

```
|— conf
|   |— layer.conf
|   |— machine
|       |— lab2-qemuarm64.conf
|— recipes-kernel
|   |— linux
|       |— files
|           |— lab2.cfg
|           |— mtd-block.cfg
|           |— yocto-testmod.patch
|       |— linux-yocto_5.10.bbappend
|       |— linux-yocto_5.14.bbappend
```


machine/lab2-qemu.conf

~/yp-summit-nov-21/poky/meta-lab2-qemuarm64/conf/machine/lab2-qemuarm64.conf

```
#@TYPE: Machine
#@NAME: lab2-qemuarm64

#@DESCRIPTION: Machine configuration for lab2-qemuarm64 systems

PREFERRED_PROVIDER_virtual/kernel ?= " linux-yocto "
PREFERRED_VERSION_linux-yocto ?= " 5.14% "
#PREFERRED_VERSION_linux-yocto ?= "5.10%"

require conf/machine/include/qemu.inc
require conf/machine/include/arm/armv8a/tune-cortexa57.inc

KERNEL_IMAGETYPE = "Image"

UBOOT_MACHINE ?= "qemu_arm64_defconfig"
...
```

linux-yocto_5.14.bbappend

~/yp-summit-nov-21/poky/meta-lab2-qemuarm64/recipes-kernel/linux/linux-yocto_5.14.bbappend

```
FILESEXTRAPATHS:prepend := "${THISDIR}/files:"

COMPATIBLE_MACHINE:lab2-qemuarm64 = "lab2-qemuarm64"

KBRANCH:lab2-qemuarm64 = " v5.14/standard/base "
KMACHINE:lab2-qemuarm64 = " qemuarm64 "

KERNEL_FEATURES:append:lab2-qemuarm64 = "  cfg/smp.scc "

#SRC_URI += "file://yocto-testmod.patch"
#SRC_URI += "file://lab2.cfg"
```

Setup our build environment

We're already in this environment from Lab#1

- `$ cd ~/yp-summit-nov-21/poky`
- `$. oe-init-build-env build-kernel/`
- `$ vim conf/local.conf`
- re-comment `#MACHINE = "lab1-qemuarm64"`
- uncomment `#MACHINE = "lab2-qemuarm64"`
- save and exit vim (`:wq`)
- `$ bitbake-layers remove-layer ../meta-lab1-qemuarm64`
- `$ bitbake-layers add-layer ../meta-lab2-qemuarm64`
- `$ cat conf/bblayers.conf`

```
/home/<user>/yp-summit-nov-21/poky/meta-lab2-qemuarm64 \
```

Lab #2 -- Build and Boot the Image

kernel compilation is ~3 minutes
kernel deploy is ~1 minute

- `bitbake core-image-base` (~6 minutes: mostly building the kernel)
- `runqemu slirp nographic`
`tmp/deploy/images/lab2-qemuarm64/Image-lab2-qemuarm64`
`.bin`
`tmp/deploy/images/lab2-qemuarm64/core-image-base-lab2`
`-qemuarm64.ext4`
 - `slirp`: user space networking (no elevated privileges required)
 - `nographic`: run in console, do not launch a GUI window
 - `path to kernel image`
 - `path to rootfs`

First boot

```
Poky (Yocto Project Reference Distro) 3.4 lab2-qemuarm64 /dev/ttyAMA0

lab2-qemuarm64 login: root
root@lab2-qemuarm64:~# uname -a
Linux lab2-qemuarm64 5.14.15-yocto-standard #1 SMP PREEMPT Fri Oct 29 01:21:02 UTC 2021 aarch64
GNU/Linux
```

yocto-testmod.patch

~/yp-summit-nov-21/poky/meta-lab2-qemuarm64/recipes-kernel/linux/files/yocto-testmod.patch

- Add module to Kconfig

```
+config YOCTO_TESTMOD
+    tristate "Yocto Test Driver"
+    help
+        This driver provides a silly message for testing Yocto.
+
```

- Add Makefile
- Add module init code (printk)

```
+#include <linux/module.h>
+
+static int __init yocto_testmod_init(void)
+{
+    printk("Krillroy swam here! ");
+
+    return 0;
+}
```

lab2.cfg

~/yp-summit-nov-21/poky/meta-lab2-qemuarm64/recipes-kernel/linux/files/lab2.cfg

- Enable our test module

```
# Enable the testmod  
CONFIG_YOCTO_TESTMOD=y
```

linux-yocto_5.14.bbappend: add patch and config

~/yp-summit-nov-21/poky/meta-lab2-qemuarm64/recipes-kernel/linux/linux-yocto_5.14.bbappend

```
FILESEXTRAPATHS:prepend := "${THISDIR}/files:"  
  
COMPATIBLE_MACHINE:lab2-qemuarm64 = "lab2-qemuarm64"  
  
KBRANCH:lab2-qemuarm64 = " v5.14/standard/base "  
KMACHINE:lab2-qemuarm64  = " qemuarm64 "  
  
KERNEL_FEATURES:append:lab2-qemuarm64 = "  cfg/smp.scc "  
  
SRC_URI += "file://yocto-testmod.patch"  
SRC_URI += "file://lab2.cfg"
```

Uncomment these two SRC_URI lines

Lab #2 -- Rebuild (only) the Kernel

```
PREFERRED_PROVIDER:virtual/kernel = "linux-yocto"
```

- `bitbake -c compile -f virtual/kernel`
 - This forces the “do_compile” task to be rerun.
- `bitbake -c deploy virtual/kernel`
 - This deploys the kernel image and artifacts to the `tmp/deploy/images/lab2-qemuarm64` directory.

```
MACHINE = "lab2-qemuarm64"
```

Lab #2 -- Boot the Image

“Up” arrow to reuse the command (it hasn’t changed) or “Ctrl-R” and start typing for bash completion.

- `runqemu slirp nographic`
`tmp/deploy/images/lab2-qemuarm64/Image-lab2-qemuarm64`
`.bin`
`tmp/deploy/images/lab2-qemuarm64/core-image-base-lab2`
`-qemuarm64.ext4`
 - `slirp`: user space networking (no elevated privileges required)
 - `nographic`: run in console, do not launch a GUI window
 - `path to kernel image`
 - `path to rootfs`

Success!

```
[    3.661508] EXT4-fs (vda): re-mounted. Opts: (null). Quota mode: disabled.
```

```
Poky (Yocto Project Reference Distro) 3.4 lab2-qemuarm64 /dev/ttyAMA0
```

```
lab2-qemuarm64 login: root
```

```
root@lab2-qemuarm64:~# dmesg | grep Krillroy
```

```
[    1.995533] Krillroy swam here!
```

```
root@lab2-qemuarm64:~#
```





Hands-on Kernel Lab #2

Exercise #2

Modify the Kernel to Make Use of an LTS Kernel Option

machine/lab2-qemu.conf: switch to 5.10 LTS kernel

~/yp-summit-nov-21/poky/meta-lab2-qemuarm64/conf/machine/lab2-qemuarm64.conf

```
#@TYPE: Machine
#@NAME: lab2-qemuarm64

#@DESCRIPTION: Machine configuration for lab2-qemuarm64 systems

PREFERRED_PROVIDER_virtual/kernel ?= " linux-yocto "
#PREFERRED_VERSION_linux-yocto ?= " 5.14%"
PREFERRED_VERSION_linux-yocto ?= " 5.10%"

require conf/machine/include/qemu.inc
require conf/machine/include/arm/armv8a/tune-cortexa57.inc

KERNEL_IMAGETYPE = "Image"

UBOOT_MACHINE ?= "qemu_arm64_defconfig"
...
```

linux-yocto_5.10.bbappend (the current LTS kernel)

~/yp-summit-nov-21/poky/meta-lab2-qemuarm64/recipes-kernel/linux/linux-yocto_5.10.bbappend

```
FILESEXTRAPATHS:prepend := "${THISDIR}/files:"

COMPATIBLE_MACHINE:lab2-qemuarm64 = "lab2-qemuarm64"

KBRANCH:lab2-qemuarm64 = " v5.10/standard/base "
KMACHINE:lab2-qemuarm64 = " qemuarm64 "

KERNEL_FEATURES:append:lab2-qemuarm64 = "  cfg/smp.scc "

#SRC_URI += "file://mtd-block.cfg"

#SRC_URI += "file://yocto-testmod.patch"
#SRC_URI += "file://lab2.cfg"
```

Lab #2 -- Rebuild (only) the Kernel

```
PREFERRED_PROVIDER:virtual/kernel = "linux-yocto"
```

- **bitbake -c compile -f virtual/kernel**
 - This forces the "do_compile" task to be rerun.
 - This time, 5.10 kernel will be built (PREFERRED_VERSION)
- **bitbake -c deploy virtual/kernel**
 - This deploys the kernel image and artifacts to the `tmp/deploy/images/lab2-qemuarm64` directory.

```
MACHINE = "lab2-qemuarm64"
```

Lab #2 -- Boot the Image

“Up” arrow to reuse the command (it hasn’t changed) or “Ctrl-R” and start typing for bash completion.

- `runqemu slirp nographic`
`tmp/deploy/images/lab2-qemuarm64/Image-lab2-qemuarm64`
`.bin`
`tmp/deploy/images/lab2-qemuarm64/core-image-base-lab2`
`-qemuarm64.ext4`
 - `slirp`: user space networking (no elevated privileges required)
 - `nographic`: run in console, do not launch a GUI window
 - `path to kernel image`
 - `path to rootfs`

Boot with LTS Kernel

```
Poky (Yocto Project Reference Distro) 3.4 lab2-qemuarm64 /dev/ttyAMA0

lab2-qemuarm64 login: root
root@lab2-qemuarm64:~# uname -a
Linux lab2-qemuarm64 5.10.76-yocto-standard #1 SMP PREEMPT Fri Oct 29 01:33:22 UTC 2021 aarch64
GNU/Linux
root@lab2-qemuarm64:~#
```

mtd-block.cfg

~/yp-summit-nov-21/poky/meta-lab2-qemuarm64/recipes-kernel/linux/files/mtd-block.cfg

- **Add Memory Technology Device block device support**

```
# Enable MTD BLOCK
CONFIG_MTD=y
CONFIG_MTD_BLOCK=y
```

linux-yocto_5.10.bbappend: add mtd-block fragment

~/yp-summit-nov-21/poky/meta-lab2-qemuarm64/recipes-kernel/linux/linux-yocto_5.10.bbappend

```
FILESEXTRAPATHS:prepend := "${THISDIR}/files:"

COMPATIBLE_MACHINE:lab2-qemuarm64 = "lab2-qemuarm64"

KBRANCH:lab2-qemuarm64 = " v5.10/standard/base "
KMACHINE:lab2-qemuarm64 = " qemuarm64"

KERNEL_FEATURES:append:lab2-qemuarm64 = "  cfg/smp.scc "

SRC_URI += "file://mtd-block.cfg"

#SRC_URI += "file://yocto-testmod.patch"
#SRC_URI += "file://lab2.cfg"
```

Lab #2 -- Rebuild (only) the Kernel

- `bitbake -c compile -f virtual/kernel`
 - This forces the “do_compile” task to be rerun.
 - This is actually now optional since the recipe is already “tainted”.
- `bitbake -c deploy virtual/kernel`
 - This deploys the kernel image and artifacts to the `tmp/deploy/images/lab2-qemuarm64` directory.

Examine the .config

linux-yocto build directory is <MACHINE>-standard-build

```
$ vim  
tmp/work/lab2_qemuarm64-poky-linux/linux-yocto/5.10.76+gitAUTOINC+e1979ceb17_be6faea8fd-r0/ linu  
x-lab2_qemuarm64-standard-build /.config
```

```
CONFIG_MTD=y
```

```
# CONFIG_MTD_TESTS is not set
```

```
...
```

```
#
```

```
# User Modules And Translation Layers
```

```
#
```

```
CONFIG_MTD_BLKDEVS=y
```

```
CONFIG_MTD_BLOCK=y
```

Search with
/CONFIG_MTD=y

A decorative pattern of overlapping hexagons in various shades of gray, located in the top-left corner of the slide.

Lab#2 Complete!

yocto ·
PROJECT

THE
LINUX
FOUNDATION



Hands-on Kernel Lab #3

Exercise #1

Working with a Custom Kernel Recipe

(When in doubt) Choosing a Kernel

<https://www.kernel.org/>



The Linux Kernel Archives

About Contact us FAQ Releases Signatures Site news

Protocol Location

HTTP <https://www.kernel.org/pub/>

GIT <https://git.kernel.org/>

RSYNC <rsync://rsync.kernel.org/pub/>

Latest Release

5.15.5

mainline:	5.16-rc2	2021-11-21	[tarball]	[patch]	[inc. patch]	[view diff]	[browse]
stable:	5.15.5	2021-11-25	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff] [browse] [changelog]
stable:	5.14.21 [EOL]	2021-11-21	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff] [browse] [changelog]
longterm:	5.10.82	2021-11-26	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff] [browse] [changelog]
longterm:	5.4.162	2021-11-26	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff] [browse] [changelog]
longterm:	4.19.218	2021-11-26	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff] [browse] [changelog]
longterm:	4.14.256	2021-11-26	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff] [browse] [changelog]
longterm:	4.9.291	2021-11-26	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff] [browse] [changelog]
longterm:	4.4.293	2021-11-26	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff] [browse] [changelog]
linux-next:	next-20211126	2021-11-26					[browse]

Other resources Social

Lab #3

- Getting to know the environment
 - ~/yp-summit-nov-21/poky/meta-lab3-qemuarm64

```
meta-lab3-qemuarm64/  
├─ conf  
│   ├── layer.conf  
│   └─ machine  
│       └─ lab3-qemuarm64.conf  
└─ recipes-kernel  
    └─ linux  
        ├── linux-yocto-custom  
        │   ├── arm64_defconfig  
        │   ├── defconfig  
        │   ├── lab3.cfg  
        │   ├── qemuarm64_defconfig  
        │   └─ yocto-testmod.patch  
        └─ linux-yocto-custom_git.bb
```

machine/lab3-qemu.conf

~/yp-summit-nov-21/poky/meta-lab3-qemuarm64/conf/machine/lab3-qemuarm64.conf

```
#@TYPE: Machine
#@NAME: lab3-qemuarm64

#@DESCRIPTION: Machine configuration for lab3-qemuarm64 systems

PREFERRED_PROVIDER_virtual/kernel ?= " linux-yocto-custom "

require conf/machine/include/qemu.inc
require conf/machine/include/arm/armv8a/tune-cortexa57.inc

KERNEL_IMAGETYPE = "Image"

UBOOT_MACHINE ?= "qemu_arm64_defconfig"

...
```

linux-yocto-custom_git.bb

~/yp-summit-nov-21/poky/meta-lab3-qemuarm64/recipes-kernel/linux/linux-yocto-custom_git.bb

```
inherit kernel
require recipes-kernel/linux/ linux-yocto.inc

# KBRANCH is the branch the used for the git clone. In this case the tip of 5.15 stable
KBRANCH = " linux-5.15.y "

SRC_URI:lab3-qemuarm64 =
"git://git.kernel.org/pub/scm/linux/kernel/git/stable/linux-stable.git;protocol=git;nocheckout=
1;branch=${KBRANCH}"
SRC_URI:lab3-qemuarm64 += "file:// defconfig "

LINUX_VERSION:lab3-qemuarm64 ?= " 5.15.2 "

# the sha of the commit for 5.15.2.  git log --grep=5.15.2 -n 1
SRCREV:lab3-qemuarm64=" 7cc36c3e14ae0af800a3a5d20cb17d0c168fc956 "

LINUX_VERSION_EXTENSION:lab3-qemuarm64 ?= " -custom "

#SRC_URI:lab3-qemuarm64 += "file://yocto-testmod.patch"
#SRC_URI:lab3-qemuarm64 += "file://lab3.cfg"
```

5.15.2 was the latest stable release
when the lab material was developed

Setup our build environment

We're already in this environment from Lab#1

- `$ cd ~/yp-summit-nov-21/poky`
- `$. oe-init-build-env build-kernel/`
- `$ vim conf/local.conf`
- re-comment `#MACHINE = "lab2-qemuarm64"`
- uncomment `#MACHINE = "lab3-qemuarm64"`
- save and exit vim (`:wq`)
- `$ bitbake-layers remove-layer ../meta-lab2-qemuarm64`
- `$ bitbake-layers add-layer ../meta-lab3-qemuarm64`
- `$ cat conf/bblayers.conf`

```
/home/<user>/yp-summit-nov-21/poky/meta-lab3-qemuarm64 \
```

Lab #3 -- Build and Boot the Image

The git clone of the kernel would have taken ~30 minutes. We did it for you already.

- `bitbake core-image-base` (~6 minutes: mostly compiling kernel)
- `runqemu slirp nographic`
`tmp/deploy/images/lab3-qemuarm64/Image-lab3-qemuarm64.bin`
`tmp/deploy/images/lab3-qemuarm64/core-image-base-lab3-qemuarm64.ext4`
 - `slirp`: user space networking (no elevated privileges required)
 - `nographic`: run in console, do not launch a GUI window
 - `path to kernel image`
 - `path to rootfs`

Booting our linux-yocto-custom kernel

```
[    3.615892] EXT4-fs (vda): re-mounted. Opts: (null). Quota mode: disabled.  
  
Poky (Yocto Project Reference Distro) 3.4 lab3-qemuarm64 /dev/ttyAMA0  
  
lab3-qemuarm64 login: root  
root@lab3-qemuarm64:~# uname -a  
Linux lab3-qemuarm64 5.15.2-custom #1 SMP PREEMPT Wed Nov 17 23:55:07 UTC 2021 aarch64  
GNU/Linux  
root@lab3-qemuarm64:~#
```

linux-yocto-custom_git.bb add patch and config

~/yp-summit-nov-21/poky/meta-lab3-qemuarm64/recipes-kernel/linux/linux-yocto-custom_git.bb

```
inherit kernel
require recipes-kernel/linux/linux-yocto.inc

# KBRANCH is the branch the used for the git clone. In this case the tip of 5.15 stable
KBRANCH = "linux-5.15.y"

SRC_URI:lab3-qemuarm64 =
"git://git.kernel.org/pub/scm/linux/kernel/git/stable/linux-stable.git;protocol=git;nocheckout=
1;branch=${KBRANCH}"
SRC_URI:lab3-qemuarm64 += "file://defconfig"

LINUX_VERSION:lab3-qemuarm64 ?= "5.15.2"

# the sha of the commit for 5.15.2. git log --grep=5.15.2 -n 1
SRCREV:lab3-qemuarm64="7cc36c3e14ae0af800a3a5d20cb17d0c168fc956"

LINUX_VERSION_EXTENSION:lab3-qemuarm64 ?= "-custom"

SRC_URI:lab3-qemuarm64 += "file://yocto-testmod.patch"
SRC_URI:lab3-qemuarm64 += "file://lab3.cfg"
```

Uncomment these two SRC_URI lines

Lab #3 -- Rebuild (only) the Kernel

```
PREFERRED_PROVIDER:virtual/kernel = "linux-yocto-custom"
```

- `bitbake -c compile -f virtual/kernel`
 - This forces the “do_compile” task to be rerun.
- `bitbake -c deploy virtual/kernel`
 - This deploys the kernel image and artifacts to the `tmp/deploy/images/lab3-qemuarm64` directory.

```
MACHINE = "lab3-qemuarm64"
```


Lab #3 -- Boot the Image

“Up” arrow to reuse the command (it hasn’t changed) or “Ctrl-R” and start typing for bash completion.

- `runqemu slirp nographic`
`tmp/deploy/images/lab3-qemuarm64/Image-lab3-qemuarm64`
`.bin`
`tmp/deploy/images/lab3-qemuarm64/core-image-base-lab3`
`-qemuarm64.ext4`
 - `slirp`: user space networking (no elevated privileges required)
 - `nographic`: run in console, do not launch a GUI window
 - `path to kernel image`
 - `path to rootfs`

Boot and look for module

```
[ 3.677484] EXT4-fs (vda): re-mounted. Opts: (null). Quota mode: disabled.
```

```
Poky (Yocto Project Reference Distro) 3.4 lab3-qemuarm64 /dev/ttyAMA0
```

```
lab3-qemuarm64 login: root
```

```
root@lab3-qemuarm64:~# dmesg | grep Billroy
```

```
root@lab3-qemuarm64:~#
```

```
root@lab3-qemuarm64:~# lsmod
```

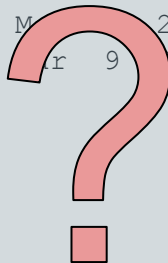
Module	Size	Used by
--------	------	---------

```
root@lab3-qemuarm64:~#
```

```
root@lab3-qemuarm64:~# ls -al /lib/modules/5.15.2-custom/kernel/
```

drwxr-xr-x	3	root	root	1024	Mar 9	2018	.
drwxr-xr-x	3	root	root	1024	Mar 9	2018	..
drwxr-xr-x	3	root	root	1024	Mar 9	2018	fs

Where is drivers/misc/yocto-testmod.ko?



machine/lab3-qemu.conf: add module to image

~/yp-summit-nov-21/poky/meta-lab3-qemuarm64/conf/machine/lab3-qemuarm64.conf

```
#@TYPE: Machine
#@NAME: lab3-qemuarm64

#@DESCRIPTION: Machine configuration for lab3-qemuarm64 systems

PREFERRED_PROVIDER_virtual/kernel ?= "linux-yocto-custom"

require conf/machine/include/qemu.inc
require conf/machine/include/arm/armv8a/tune-cortexa57.inc

KERNEL_IMAGETYPE = "Image"

UBOOT_MACHINE ?= "qemu_arm64_defconfig"

...
```

Uncomment the MACHINE_ESSENTIAL_EXTRA_RRECOMMENDS line at the end of the file

```
MACHINE_ESSENTIAL_EXTRA_RRECOMMENDS += " kernel-module-yocto-testmod"
```

linux-yocto-custom_git.bb deploy and autoload module

~/yp-summit-nov-21/poky/meta-lab3-qemuarm64/recipes-kernel/linux/linux-yocto-custom_git.bb

```
# KBRANCH is the branch the used for the git clone. In this case the tip of 5.15 stable
KBRANCH = "linux-5.15.y"

SRC_URI:lab3-qemuarm64 =
"git://git.kernel.org/pub/scm/linux/kernel/git/stable/linux-stable.git;protocol=git;nocheckout=
1;branch=${KBRANCH}"
SRC_URI:lab3-qemuarm64 += "file://defconfig"

LINUX_VERSION:lab3-qemuarm64 ?= "5.15.2"

# the sha of the commit for 5.15.2.  git log --grep=5.15.2 -n 1
SRCREV:lab3-qemuarm64="7cc36c3e14ae0af800a3a5d20cb17d0c168fc956"

LINUX_VERSION_EXTENSION:lab3-qemuarm64 ?= "-custom"

SRC_URI:lab3-qemuarm64 += "file://yocto-testmod.patch"
SRC_URI:lab3-qemuarm64 += "file://lab3.cfg"

KERNEL_MODULE_AUTOLOAD += "yocto-testmod"
```

Uncomment the KERNEL_MODULE_AUTOLOAD line

Fix yocto-testmod output (Try it yourself without this)

~/yp-summit-nov-21/poky/meta-lab3-qemuarm64/recipes-kernel/linux/linux-yocto-custom/yocto-testmod.patch

```
+#include <linux/module.h>
+
+static int __init yocto_testmod_init(void)
+{
+    printk("Billroy quacked here! \n");
+
+    return 0;
+}
+
+static void __exit yocto_testmod_exit(void)
+{
+    printk("Billroy did not quack here! \n");
+}
+
+module_init(yocto_testmod_init);
+module_exit(yocto_testmod_exit);
```

Terminate the string with \n

Terminate the string with \n

Lab #3 -- Rebuild and Reboot the Image

We changed the image metadata, not just the kernel this time.

- `bitbake core-image-base`

- `runqemu slirp nographic`

`tmp/deploy/images/lab3-qemuarm64/Image-lab3-qemuarm64
.bin`

`tmp/deploy/images/lab3-qemuarm64/core-image-base-lab3
-qemuarm64.ext4`

- `slirp`: user space networking (no elevated privileges required)
- `nographic`: run in console, do not launch a GUI window
- `path to kernel image`
- `path to rootfs`

Reboot and look for module again (SUCCESS!)

```
[ 4.765660] EXT4-fs (vda): re-mounted. Opts: (null). Quota mode: disabled.  
[ 5.107150] Billroy quacked here!
```

Poky (Yocto Project Reference Distro) 3.4 lab3-qemuarm64 /dev/ttyAMA0

lab3-qemuarm64 login: root

root@lab3-qemuarm64:~# **lsmod**

Module	Size	Used by
yocto_testmod	16384	0

root@lab3-qemuarm64:~# **ls -la /lib/modules/5.15.2-custom/kernel/drivers/misc/**

drwxr-xr-x	2	root	root	1024	Mar 9 2018	.
drwxr-xr-x	3	root	root	1024	Mar 9 2018	..
-rw-r--r--	1	root	root	4368	Mar 9 2018	yocto-testmod.ko

root@lab3-qemuarm64:~# **cat /etc/modules-load.d/yocto-testmod.conf**

yocto-testmod

root@lab3-qemuarm64:~# **rmmod yocto_testmod**

```
[ 64.045745] Billroy did not quack here!
```

root@lab3-qemuarm64:~#

lol

Without the `\n` fix in `yocto-testmod.patch`, we would not see the “Billroy” messages.



Lab#3 Complete!

yocto ·
PROJECT

 THE
LINUX
FOUNDATION



Hands-on Kernel Lab #4

Exercise #1

Working with local git tree and out-of-tree module

Lab #4

- Getting to know the environment
 - ~/yp-summit-nov-21/poky/meta-lab4-qemuarm64

```
meta-lab4-qemuarm64/  
├── conf  
│   ├── layer.conf  
│   └── machine  
│       └── lab4-qemuarm64.conf  
└── recipes-kernel  
    ├── hello-mod  
    │   ├── files  
    │   │   ├── COPYING  
    │   │   ├── Makefile  
    │   │   └── hello.c  
    │   └── hello-mod_0.1.bb  
    └── linux  
        ├── linux-yocto-custom  
        │   └── defconfig  
        └── linux-yocto-custom_git.bb
```

machine/lab4-qemu.conf

~/yp-summit-nov-21/poky/meta-lab4-qemuarm64/conf/machine/lab4-qemuarm64.conf

```
#@TYPE: Machine
#@NAME: lab4-qemuarm64

#@DESCRIPTION: Machine configuration for lab4-qemuarm64 systems

PREFERRED_PROVIDER_virtual/kernel ?= " linux-yocto-custom "

require conf/machine/include/qemu.inc
require conf/machine/include/arm/armv8a/tune-cortexa57.inc

KERNEL_IMAGETYPE = "Image"

UBOOT_MACHINE ?= "qemu_arm64_defconfig"

...
```

linux-yocto-custom_git.bb

~/yp-summit-nov-21/poky/meta-lab3-qemuarm64/recipes-kernel/linux/linux-yocto-custom_git.bb

```
inherit kernel
require recipes-kernel/linux/linux-yocto.inc

SRC_URI:lab4-qemuarm64 =
    "git://${HOME}/yp-summit-nov-21/linux-stable-work.git;protocol=file;name=machine;branch=${KBRANCH}"
SRC_URI:lab4-qemuarm64 +=
    "git://git.yoctoproject.org/yocto-kernel-cache;type=kmeta;name=meta;branch=yocto-5.15;destsuffix=${KMETA}"
SRC_URI:lab4-qemuarm64 += "file:// defconfig"

KBRANCH = "work-branch"
KMETA = "kernel-meta"

LINUX_VERSION:lab4-qemuarm64 ?= "5.15.2"
LINUX_VERSION_EXTENSION:lab4-qemuarm64 ?= "-custom"

SRCREV_machine:lab4-qemuarm64 = " ${AUTOREV} "
SRCREV_meta:lab4-qemuarm64 = " bee5d6a15909f05935f4a61f83a72cddfca7934a "
```

Our local working git tree

5.15.2 was the latest stable release
when the lab material was developed

Setup our build environment

We're already in this environment from Lab#1

- `$ cd ~/yp-summit-nov-21/poky`
- `$. oe-init-build-env build-kernel/`
- `$ vim conf/local.conf`
- re-comment `#MACHINE = "lab3-qemuarm64"`
- uncomment `#MACHINE = "lab4-qemuarm64"`
- save and exit vim (`:wq`)
- `$ bitbake-layers remove-layer ../meta-lab3-qemuarm64`
- `$ bitbake-layers add-layer ../meta-lab4-qemuarm64`
- `$ cat conf/bblayers.conf`

```
/home/<user>/yp-summit-nov-21/poky/meta-lab4-qemuarm64 \
```

Lab #4 -- Clone our working git tree

- `pushd ~/yp-summit-nov-21`
- `git clone -b v5.15.2`
`poky/downloads/git2/git.kernel.org.pub.scm.linux`
`.kernel.git.stable.linux-stable.git`
`linux-stable-work.git`
- `cd linux-stable-work.git/`
- `git checkout -b work-branch`
- `popd`
- You should be in `~/yp-summit-nov-21/poky/build-kernel`

Re-use download already
done for you.

Lab #4 -- Build and Boot the Image

- `bitbake core-image-base` (~6 minutes: mostly compiling kernel)
- `runqemu slirp nographic`
`tmp/deploy/images/lab4-qemuarm64/Image-lab4-qemuarm64`
`.bin`
`tmp/deploy/images/lab4-qemuarm64/core-image-base-lab4`
`-qemuarm64.ext4`
 - `slirp`: user space networking (no elevated privileges required)
 - `nographic`: run in console, do not launch a GUI window
 - `path to kernel image`
 - `path to rootfs`

Booting our linux-yocto-custom kernel

```
[    3.676603] EXT4-fs (vda): re-mounted. Opts: (null). Quota mode: disabled.  
  
Poky (Yocto Project Reference Distro) 3.4 lab4-qemuarm64 /dev/ttyAMA0  
  
lab4-qemuarm64 login: root  
root@lab4-qemuarm64:~# uname -a  
Linux lab4-qemuarm64 5.15.2-custom #1 SMP PREEMPT Fri Nov 12 14:05:52 UTC 2021 aarch64  
GNU/Linux  
root@lab4-qemuarm64:~#
```


hello-mod_0.1.bb

~/yp-summit-nov-21/poky/meta-lab4-qemuarm64/recipes-kernel/hello-mod/hello-mod_0.1.bb

```
DESCRIPTION = "hello-world-mod tests the module.bbclass mechanism."  
LICENSE = "GPLv2"  
LIC_FILES_CHKSUM = "file://COPYING;md5=12f884d2ae1ff87c09e5b7ccc2c4ca7e"  
  
inherit module  
  
PR = "r0"  
PV = "0.1"  
  
SRC_URI = "file:// Makefile \  
           file:// hello.c \  
           file:// COPYING \  
           "  
  
S = "${WORKDIR}"
```

module.bbclass expects a
Makefile, code and a license file

hello.c

~/yp-summit-nov-21/poky/meta-lab4-qemuarm64/recipes-kernel/hello-mod/files/hello.c

```
#include <linux/module.h>

int init_module(void)
{
    printk("Hello World!\n");
    return 0;
}

void cleanup_module(void)
{
    printk("Goodbye Cruel World!\n");
}

MODULE_LICENSE("GPL");
```

Makefile

~/yp-summit-nov-21/poky/meta-lab4-qemuarm64/recipes-kernel/hello-mod/files/Makefile

```
obj-m := hello.o

SRC := $(shell pwd)

all:
    $(MAKE) -C $(KERNEL_SRC) M=$(SRC)

modules_install:
    $(MAKE) -C $(KERNEL_SRC) M=$(SRC) modules_install

clean:
    rm -f *.o *~ core .depend *.cmd *.ko *.mod.c
    rm -f Module.markers Module.symvers modules.order
    rm -rf .tmp_versions Modules.symvers
```

machine/lab4-qemu.conf: add module to image

~/yp-summit-nov-21/poky/meta-lab4-qemuarm64/conf/machine/lab4-qemuarm64.conf

```
#@TYPE: Machine
#@NAME: lab4-qemuarm64

#@DESCRIPTION: Machine configuration for lab4-qemuarm64 systems

PREFERRED_PROVIDER_virtual/kernel ?= "linux-yocto-custom"

require conf/machine/include/qemu.inc
require conf/machine/include/arm/armv8a/tune-cortexa57.inc

KERNEL_IMAGETYPE = "Image"

UBOOT_MACHINE ?= "qemu_arm64_defconfig"

...
```

Uncomment the MACHINE_ESSENTIAL_EXTRA_RRECOMMENDS line at the end of the file

```
MACHINE_ESSENTIAL_EXTRA_RRECOMMENDS += " hello-mod "
```

not kernel-module-hello-mod because it is a standalone recipe, not built with linux-yocto-custom

Lab #4 -- Rebuild and Reboot the Image

We changed the image metadata, not the kernel this time.

- `bitbake core-image-base`

- `runqemu slirp nographic`

`tmp/deploy/images/lab4-qemuarm64/Image-lab4-qemuarm64
.bin`

`tmp/deploy/images/lab4-qemuarm64/core-image-base-lab4
-qemuarm64.ext4`

- `slirp`: user space networking (no elevated privileges required)
- `nographic`: run in console, do not launch a GUI window
- `path to kernel image`
- `path to rootfs`

Reboot and look for module (SUCCESS!)

```
[ 4.755917] EXT4-fs (vda): re-mounted. Opts: (null). Quota mode: disabled.
```

```
Poky (Yocto Project Reference Distro) 3.4 lab4-qemuarm64 /dev/ttyAMA0
```

```
lab4-qemuarm64 login: root
```

```
root@lab4-qemuarm64:~# ls -la /lib/modules/5.15.2-custom/extra/
```

```
drwxr-xr-x  2 root    root          1024 Mar  9  2018 .
drwxr-xr-x  4 root    root          1024 Mar  9  2018 ..
-rw-r--r--  1 root    root         3880 Mar  9  2018 hello.ko
```

```
root@lab4-qemuarm64:~# modprobe hello
```

```
[ 36.197006] hello: loading out-of-tree module taints kernel.
```

```
[ 36.204125] Hello World!
```

```
root@lab4-qemuarm64:~# lsmod
```

Module	Size	Used by
hello	16384	0

```
root@lab4-qemuarm64:~# rmmod hello
```

```
[ 50.199061] Goodbye Cruel World!
```

```
root@lab4-qemuarm64:~#
```

vo!



Hands-on Kernel Lab #4

Exercise #2

Modifying kernel in local git tree

Lab #4 -- Modifying our local kernel

- `$ pushd ~/yp-summit-nov-21`
- `$ cd linux-stable-work.git/`
- `$ vim fs/filesystems.c`

fs/filesystems.c: add a printk statement

~/yp-summit-nov-21/linux-stable-work.git/fs/filesystems.c

```
#ifdef CONFIG_PROC_FS
static int filesystems_proc_show(struct seq_file *m, void *v)
{
    struct file_system_type * tmp;

    read_lock(&file_systems_lock);
    tmp = file_systems;
    while (tmp) {
        seq_printf(m, "%s\t%s\n",
                    (tmp->fs_flags & FS_REQUIRES_DEV) ? "" : "nodev",
                    tmp->name);
        tmp = tmp->next;
    }
    read_unlock(&file_systems_lock);

    printk("Kilfoy was here!\n");

    return 0;
}
```

Insert this line

Verify the changes with `git diff -p HEAD`

```
$ git diff -p HEAD
diff --git a/fs/filesystems.c b/fs/filesystems.c
index 58b9067b2391..8229ab25c7d1 100644
--- a/fs/filesystems.c
+++ b/fs/filesystems.c
@@ -247,6 +247,9 @@ static int filesystems_proc_show(struct seq_file *m, void *v)
     tmp = tmp->next;
 }
 read_unlock(&file_systems_lock);
+
+ printk("Kilfoy was here!\n");
+
 return 0;
 }
```

Lab #4 -- Commit our change

- `$ git config --global user.email "you@example.com"`
- `$ git config --global user.name "Your Name"`
- `$ git commit -a -m "fs/filesystems.c: add a message that will be logged to the kernel log when you 'cat /proc/filesystems'."`
`[work-branch b1e9c0df483e] fs/filesystems.c: add a message that will be logged to the kernel log when you 'cat /proc/filesystems'.`
- `$ git log`

Verify the commit with `git log`

```
commit ble9c0df483e353ac821be21bbf3c2a547130eca (HEAD -> work-branch)
Author: Your Name <you@example.com>
Date:   Sat Nov 27 17:22:15 2021 +0000

    fs/filesystems.c: add a message that will be logged to the kernel log when you 'cat
    /proc/filesystems'.

commit 7cc36c3e14ae0af800a3a5d20cb17d0c168fc956 (tag: v5.15.2)
Author: Greg Kroah-Hartman <gregkh@linuxfoundation.org>
Date:   Fri Nov 12 15:05:52 2021 +0100

    Linux 5.15.2

Link: https://lore.kernel.org/r/20211110182003.700594531@linuxfoundation.org
Tested-by: Florian Fainelli <f.fainelli@gmail.com>
Tested-by: Linux Kernel Functional Testing <lkft@linaro.org>
Tested-by: Shuah Khan <skhan@linuxfoundation.org>
Tested-by: Fox Chen <foxhlchen@gmail.com>
Tested-by: Salvatore Bonaccorso <carnil@debian.org>
...
```

q to quit

Lab #4 -- Build our modified kernel

- `$ popd`
- You should be in `~/yp-summit-nov-21/poky/build-kernel`

We need to checkout our modified source

- `$ bitbake -c cleanall virtual/kernel`
- `$ bitbake -c deploy virtual/kernel`

Lab #4 -- Boot the Image

“Up” arrow to reuse the command (it hasn’t changed) or “Ctrl-R” and start typing for bash completion.

- `runqemu slirp nographic`
`tmp/deploy/images/lab4-qemuarm64/Image-lab4-qemuarm64`
`.bin`
`tmp/deploy/images/lab4-qemuarm64/core-image-base-lab4`
`-qemuarm64.ext4`
 - `slirp`: user space networking (no elevated privileges required)
 - `nographic`: run in console, do not launch a GUI window
 - `path to kernel image`
 - `path to rootfs`

Trigger our change with `cat /proc/filesystems`

```
[ 3.630161] EXT4-fs (vda): re-mounted. Opts: (null). Quota mode: disabled.  
[ 3.698557] Kilfoy was here!  
[ 3.705723] Kilfoy was here!
```

every time /proc/filesystems is read by other processes we see our printk message

Poky (Yocto Project Reference Distro) 3.4 lab4-qemuarm64 /dev/ttyAMA0

```
lab4-qemuarm64 login: root  
root@lab4-qemuarm64:~# cat /proc/filesystems  
[ 17.605165] Kilfoy was here!  
nodev      sysfs  
nodev      tmpfs  
...
```

Wow



Hands-on Kernel Lab #4

Exercise #3

Using linux-yocto kernel in local git tree

Lab #4 -- Clone our working linux-yocto git tree

- `$ pushd ~/yp-summit-nov-21`

- `$ git clone -b v5.10`

Re-use download already
done for you.

`poky/downloads/git2/git.yoctoproject.org.linux-y
octo.git linux-yocto-5.10.git`

- `$ cd linux-yocto-5.10.git/`

- `$ git checkout v5.10/standard/base`

- `$ popd`

- You should be in `~/yp-summit-nov-21/poky/build-kernel`

Switch to lab2 environment

We're already in this environment from Lab#1

- `$ cd ~/yp-summit-nov-21/poky`
- `$. oe-init-build-env build-kernel/`
- `$ vim conf/local.conf`
- re-comment `#MACHINE = "lab4-qemuarm64"`
- uncomment `#MACHINE = "lab2-qemuarm64"`
- save and exit vim (`:wq`)
- `$ bitbake-layers remove-layer ../meta-lab4-qemuarm64`
- `$ bitbake-layers add-layer ../meta-lab2-qemuarm64`
- `$ cat conf/bblayers.conf`

```
/home/<user>/yp-summit-nov-21/poky/meta-lab2-qemuarm64 \
```

linux-yocto_5.10.bbappend: point to our local git tree

~/yp-summit-nov-21/poky/meta-lab2-qemuarm64/recipes-kernel/linux/linux-yocto_5.10.bbappend

```
FILESEXTRAPATHS:prepend := "${THISDIR}/files:"

COMPATIBLE_MACHINE:lab2-qemuarm64 = "lab2-qemuarm64"

SRC_URI = "git://${HOME}/yp-summit-nov-21/linux-yocto-5.10.git;protocol=file;name=machine;branch=${KBRANCH};\
\
git://git.yoctoproject.org/yocto-kernel-cache;type=kmeta;name=meta;branch=yocto-5.10;destsuffix=${KMETA}"
KERNEL_VERSION_SANITY_SKIP="1"

SRCREV_machine:pn-linux-yocto:lab2-qemuarm64 ?= "${AUTOREV}"
SRCREV_meta:pn-linux-yocto:lab2-qemuarm64 ?= "${AUTOREV}"

KBRANCH:lab2-qemuarm64 = "v5.10/standard/base"
KMACHINE:lab2-qemuarm64 = "qemuarm64"

KERNEL_FEATURES:append:lab2-qemuarm64 = " cfg/smp.scc"

SRC_URI += "file://mtd-block.cfg"

#SRC_URI += "file://yocto-testmod.patch"
#SRC_URI += "file://lab2.cfg"
```

machine/lab2-qemu.conf: verify 5.10 LTS kernel

~/yp-summit-nov-21/poky/meta-lab2-qemuarm64/conf/machine/lab2-qemuarm64.conf

```
#@TYPE: Machine
#@NAME: lab2-qemuarm64

#@DESCRIPTION: Machine configuration for lab2-qemuarm64 systems

PREFERRED_PROVIDER_virtual/kernel ?= "linux-yocto"
#PREFERRED_VERSION_linux-yocto ?= " 5.14%"
PREFERRED_VERSION_linux-yocto ?= " 5.10%"

require conf/machine/include/qemu.inc
require conf/machine/include/arm/armv8a/tune-cortexa57.inc

KERNEL_IMAGETYPE = "Image"

UBOOT_MACHINE ?= "qemu_arm64_defconfig"
...
```

Rebuild the Kernel and Reboot the Image

- `bitbake -c deploy virtual/kernel`
- `runqemu slirp nographic`
`tmp/deploy/images/lab2-qemuarm64/Image-lab2-qemuarm64`
`.bin`
`tmp/deploy/images/lab2-qemuarm64/core-image-base-lab2`
`-qemuarm64.ext4`
 - `slirp`: user space networking (no elevated privileges required)
 - `nographic`: run in console, do not launch a GUI window
 - `path to kernel image`
 - `path to rootfs`

Boot our local linux-yocto LTS Kernel

```
[    3.642731] EXT4-fs (vda): re-mounted. Opts: (null)

Poky (Yocto Project Reference Distro) 3.4 lab2-qemuarm64 /dev/ttyAMA0

lab2-qemuarm64 login: root
root@lab2-qemuarm64:~# uname -a
Linux lab2-qemuarm64 5.10.76-yocto-standard #1 SMP PREEMPT Fri Oct 29 01:33:22 UTC 2021 aarch64
GNU/Linux
root@lab2-qemuarm64:~#
```



Hands-on Kernel Lab #4

Exercise #4

Modifying linux-yocto kernel in local git tree

Lab #4 -- Modifying our local linux-yocto kernel

- `$ pushd ~/yp-summit-nov-21`
- `$ cd linux-yocto-5.10.git/`
- `$ vim fs/filesystems.c`

fs/filesystems.c: add a printk statement

~/yp-summit-nov-21/linux-yocto-5.10.git/fs/filesystems.c

```
#ifdef CONFIG_PROC_FS
static int filesystems_proc_show( struct seq_file *m, void *v)
{
    struct file_system_type * tmp;

    read_lock(&file_systems_lock);
    tmp = file_systems;
    while (tmp) {
        seq_printf(m, "%s\t%s\n",
                    (tmp->fs_flags & FS_REQUIRES_DEV) ? "" : "nodev",
                    tmp->name);
        tmp = tmp->next;
    }
    read_unlock(&file_systems_lock);

    printk("Robroy drank here!\n");

    return 0;
}
```

Search quickly with
/filesystems_proc_show
(

Insert this line

Verify the changes with `git diff -p HEAD`

```
$ git diff -p HEAD
diff --git a/fs/filesystems.c b/fs/filesystems.c
index 90b8d879fbaf..ae166f9240b8 100644
--- a/fs/filesystems.c
+++ b/fs/filesystems.c
@@ -240,6 +240,9 @@ static int filesystems_proc_show(struct seq_file *m, void *v)
     tmp = tmp->next;
 }
 read_unlock(&file_systems_lock);
+
+ printk("Robroy drank here!\n");
+
 return 0;
 }
```

Commit our change

- `$ git config --global user.email "you@example.com"`
- `$ git config --global user.name "Your Name"`
- `$ git commit -a -m "fs/filesystems.c: add a message that will be logged to the kernel log when you 'cat /proc/filesystems'."`
`[v5.10/standard/base 83cc48ca6565]`
`fs/filesystems.c: add a message that will be logged to the kernel log when you 'cat /proc/filesystems'.`
`1 file changed, 3 insertions(+)`
- `$ git log`

Verify the commit with `git log v5.10/standard/base`

```
commit 83cc48ca6565b24c40104d04515e70fbd1d6c317 (HEAD -> v5.10/standard/base)
Author: Your Name <you@example.com>
Date: Sat Nov 27 19:06:49 2021 +0000

    fs/filesystems.c: add a message that will be logged to the kernel log when you 'cat
    /proc/filesystems'.

commit 2daa192783edd4974da8e900c0dc93186e57a838 (origin/v5.10/standard/base)
Merge: 789500dfe326 5040520482a5
Author: Bruce Ashfield <bruce.ashfield@gmail.com>
Date: Tue Nov 9 11:36:52 2021 -0500

    Merge tag 'v5.10.78' into v5.10/standard/base

    This is the 5.10.78 stable release

    # gpg: Signature made Sat 06 Nov 2021 09:10:26 AM EDT
    # gpg: using RSA key 647F28654894E3BD457199BE38DBBDC86092693E
    # gpg: Can't check signature: No public key

commit 5040520482a594e92d4f69141229a6dd26173511 (tag: v5.10.78, origin/v5.10/base)
Author: Greg Kroah-Hartman <gregkh@linuxfoundation.org>
```

q to quit

Build our modified kernel

- `$ popd`
- You should be in `~/yp-summit-nov-21/poky/build-kernel`

We need to checkout our modified source

- `$ bitbake -c cleanall virtual/kernel`
- `$ bitbake -c deploy virtual/kernel`

NOTE: If the syntax of the **SRCREV** `${AUTOREV}` statements is correct, you should see the commit hash from our change in the build:

```
$ ls tmp/work/lab2_qemuarm64-poky-linux/linux-yocto/  
5.10.76+gitAUTOINC+d17e9fb459_83cc48ca65-r0
```

Reboot the Image

- `runqemu slirp nographic`
`tmp/deploy/images/lab2-qemuarm64/Image-lab2-qemuarm64`
`.bin`
`tmp/deploy/images/lab2-qemuarm64/core-image-base-lab2`
`-qemuarm64.ext4`
 - `slirp`: user space networking (no elevated privileges required)
 - `nographic`: run in console, do not launch a GUI window
 - `path to kernel image`
 - `path to rootfs`

Trigger our change with `cat /proc/filesystems`

```
[ 3.662291] EXT4-fs (vda): re-mounted. Opts: (null)
[ 3.730359] Robroy drank here!
[ 3.737545] Robroy drank here!
```

every time /proc/filesystems is read by other processes we see our printk message

Poky (Yocto Project Reference Distro) 3.4 lab2-qemuarm64 /dev/ttyAMA0

lab2-qemuarm64 login: root

root@lab2-qemuarm64:~# **cat /proc/filesystems**

```
[ 17.080748] Robroy drank here!
```

```
nodev      sysfs
nodev      tmpfs
nodev      bdev
nodev      proc
nodev      cgroup
...
```

Wow

A decorative pattern of overlapping hexagons in various shades of gray, located in the top-left corner of the slide.

Lab#4 Complete!

yocto ·
PROJECT

THE
LINUX
FOUNDATION



Achievement Unlocked!

yocto ·
PROJECT

THE
LINUX
FOUNDATION

What is the Yocto Project® ?

**IT'S NOT AN EMBEDDED LINUX DISTRIBUTION,
IT CREATES A CUSTOM ONE FOR YOU.**



The Yocto Project (YP) is an open source collaboration project that helps developers create custom Linux-based systems regardless of the hardware architecture.

The project provides a flexible set of tools and a space where embedded developers worldwide can share technologies, software stacks, configurations, and best practices that can be used to create tailored Linux images for embedded and IOT devices, or anywhere a customized Linux OS is needed.



yocto ·
PROJECT

THE
LINUX
FOUNDATION